4.2 答：栈和队列都是线性表，但是是受限的线性表，对插入、删除运算加以限制。栈是只允许在一端进行插入、删除运算，是后进先出表；而队列是只允许在一端进行插入、另一端进行删除运算，是先进先出表。

4.3 答：栈的入栈、出栈操作均在栈顶进行，栈顶指针指向栈顶元素处。入栈操作先将栈顶指针加 1，然后将入栈元素放到栈顶指针所指示的位置上。出栈操作先从栈顶指针指向位置，然后将栈顶指针减 1

**4.4 补充题：**

假设以带头结点的循环链表表示队列，并且只设一个指针指向队尾元素结点(**注意不设头指针**)，试编写相应的队列初始化、入队列和出队列的算法。

解：存储表示：

typedef Struct CQNode {

CQElemType data;

Struct CQNode \*next;

}CQNode;

typedef Struct {

CQueuePtr rear;

}CLinkQueue;

算法：

void InitCiQueue (CLinkQueue &Q) { //初始化循环链表表示的队列Q  
  Q.rear =(CQNode\*)malloc(sizeof(CQNode));

if (!Q.rear) exit (OVERFLOW) ;  
  Q.rear ->next= Q.rear;  
} //InitCiQueue

void EnCQueue(CLinkQueue &Q,int x) { //把元素x插入循环链表表示的队列Q,Q指向

队尾元素,Q.rear->next指向头结点,Q.rear->next->next指向队头元素  
 p=(CQNode\*)malloc(sizeof(CQNode));  
 if (!p) exit (OVERFLOW);  
  p->data=x;  
  p->next= Q.rear ->next; //直接把p加在Q的后面  
  Q.rear ->next=p;  
  Q.rear =p;   //修改尾指针  
} // EnCQueue

Status DeCQueue(CLinkQueue &Q,int x) {

//从循环链表表示的队列Q头部删除元素x

  if(Q.rear==Q.rear->next)

return INFEASIBLE; //队列已空

p= Q.rear->next->next;

   x=p->data;

   Q.rear->next->next=p->next;

   if (Q.rear==p) Q.rear=Q.rear->next;

free(p);

 return OK;

}//DeCQueue

